Vol.2022-MUS-134 No.12
Vol.2022-SLP-142 No.12
2022/6/17

量子コンピュータによるメロディ生成の試み ー遺伝的アルゴリズムを用いた量子回路デザインー

平井 辰典[1,a)]

概要：音楽生成はコンピュータの黎明期から取り組まれてきた研究課題である．量子コンピュータの実現を視野に入れた研究開発が進む中で，本稿では量子コンピュータの音楽生成への応用の可能性について議論する．本稿では，量子コンピュータの中でも，特に量子ゲートによって構成される量子回路を用いたメロディ生成の手法について検討し，その初期試行について述べる．具体的には，任意の入力音に対して後続音を生成するような量子回路を構成する量子ゲートの組み合わせを遺伝的アルゴリズムによってデザインする手法を提案する．特定の楽曲を学習することでデザインした量子回路を用いて，新たなメロディの生成を試みる．

1. はじめに

1957 年に ILLIAC I によって生成された「イリアック組曲」は，世界で初めてコンピュータによって自動作曲された楽曲である．コンピュータの黎明期であった当時から現在に至るまで，自動作曲に代表される音楽生成の技術は多くの研究者によって長期間に渡り研究され続けている研究課題である．

近年，次世代のコンピュータとして期待されている量子コンピュータは，その実現が徐々に現実味を帯びつつある状況にある．2022 年 5 月現在では，ノイズによってエラーが起こることが前提の NISQ（Noisy Intermediate-Scale Quantum Computer）という量子コンピュータが数十〜百量子ビット程度の規模で実現しており，クラウド上で利用できるような状況にある．クラウド上で利用できる量子コンピュータである IBM Q を開発している IBM 社は，2025 年までに 4000 量子ビット以上の量子コンピュータの開発を目指すという計画を発表している [1].

このような，次世代のコンピュータの実現にあたって，そのアプリケーションの可能性を追求することは量子コンピュータの必要性を議論する上で重要である．なぜなら，現状では量子コンピュータはまだどのような分野にどのくらい役に立つのかが正確にはわかっておらず，古典コンピュータよりも優れたコンピュータの形態と言えるのかどうかも不明瞭な状況だからである，量子コンピュータの応用が期待される問題として，量子化学計算，組み合わせ最適化問題，金融シミュレーション，量子機械学習などが挙げられているが，その他の諸問題に対しても未知の応用先は多くあると予想される．一方で，現在情報科学分野で研究されている研究課題の多くは，必ずしも量子コンピュータの導入が必要なものばかりではなく，むしろ古典コンピュータのみで取り組んだ方が正確かつ効率的であるようなタスクがほとんどを占めていると考えられる．

本稿では，長年，従来の古典コンピュータで研究され続けてきた音楽生成のタスクに関して，仮に量子コンピュータを応用する場合には，どのような応用ができるのかについて検討する，具体的には，現状のゲート式量子コンピュータでどのように音楽生成が実現可能なのかについて検討し，その初期試行として，遺伝的アルゴリズムを用いた量子回路デザインの手法を提案する．

2. 量子ビットとゲート式量子コンピュータ

量子コンピュータは，量子の重ね合わせ状態を表現可能な量子ビット（qubit）による演算を行うコンピュータである．現在普及している古典コンピュータで扱われているビット（古典ビット）は量子コンピューティングの文脈では古典ビットと呼ばれ，1 つのビットが常に 0 か 1 のどちらかの状態を取る．一方で量子ビットは，0 と 1 の両方が重ね合わされた状態を表現可能である．例えば，2 量子ビットは，「00」,「01」,「10」,「11」の 4 通りの状態を重ね合わせた状態で存在することができる．

この量子ビットの性質をうまく活用することで，一度の処理で多くの入力の組み合わせに関する演算を行うことが

[1] 駒澤大学
Komazawa University
[a)] thirai@komazawa-u.ac.jp

でき，巧妙な量子アルゴリズムを組むことによって求める答えに効率的にたどりつくことも可能である．しかし，量子ビットは，測定を行うことで 0 か 1 のどちらかの状態に確定してしまうという性質を持っている．そのため，必ず所望の結果が得られるような量子回路（もしくは量子アルゴリズム）を設計することは困難であり，実際には複数回の実行結果を基に出力の偏りから結果を確率的に判断することになる．

現状の量子コンピュータのプログラミングは，量子ビットに対する種々の演算を実現する量子ゲートを組み合わせて量子回路を作成するゲート式量子コンピュータに対するものが主流である．IBM 社が提供する IBM Q で動作するフレームワークである Qiskit を使ったプログラミングが現時点での代表的な量子プログラミングの例である．Qiskit は Python のフレームワークであり，プログラム自体は Python によって記述するが，Qiskit で記述する命令の内容は量子ゲートの組み合わせによる量子回路の作成やその上での量子計算のシミュレーション等に関するものである．

Qiskit を用いて行う量子回路のデザインは，現状の量子プログラミングにおける難しい点で，重ね合わせ状態に関する複雑な演算を組み合わせる必要があるため，量子計算に関する知識が不可欠となる．また，問題をさらに難しくする要因として，量子の重ね合わせ状態が確率的に記述されるという点がある．古典ビットでは，エラーの影響を考えなければ 0 という状態は常に 0 であり，1 という状態は常に 1 である．一方で，量子ビットの重ね合わせ状態の場合，同じ量子回路で処理を行っても，測定を行うことで重ね合わせ状態が壊れて（収束して）しまい，0 と 1 のどちらかが確率的に測定される．そのため，同じ量子回路を用いても結果が常に同じになるわけではない．さらに，現状の NISQ では量子回路を構成するゲートの構成が複雑になればなるほどエラー率も上がり，誤った計算結果が返されてしまうことも多い．このような，原因と結果の関係の煩雑さも量子コンピュータの活用に際しての障壁となっていると考えられる．

3. 関連研究

量子コンピューティングを音楽表現に応用する試みはこれまでにも幾度か試行されており，このような試みは Quantum Music（量子音楽）と呼ばれている．Kirke らが提案した Q-MUSE は，奏者が装着したボタンコントローラとジェスチャーコントローラによって量子コンピュータに入力されるパラメータを変化させることで出力音が変化するようなライブパフォーマンスを想定した音楽システムである [2]．Clemente らは，音に関するパラメータを量子回路によって制御するようなキーボードである “qeyboard” を提案している [3].

Weimer の Listen to Quantum Computer Music では，有名な量子アルゴリズムである Grover のアルゴリズムや Shor のアルゴリズムを構成する量子ゲートを音符に変換することで音楽を生成することを試みた [4]．量子回路の回路図は左から右にゲートが配置された見た目から，楽譜に例えられることがあるが，Weimer は実際に量子回路を楽譜に見立てて音楽を生成した．これは，量子アルゴリズムによって音楽を生成する試みではなく，量子回路そのものを楽譜に変換するという試みであり，量子音楽とは異なるものであると捉えられる．

QuTune Project[5] は量子コンピューティングによる音楽制作を目的とした研究プロジェクトであり，このグループは 2021 年に量子音楽に関する初めての国際シンポジウムである ISQCMC[*1] を開催している．QuTune のグループでは，いくつかの論文や総説を発表している．これまでに発表された総説 [6]，[7] では，量子コンピュータとコンピュータ音楽の基礎についての解説から逆 FFT を使った Quantum Vocal Synthesizer，や Quantum Walk Sequencer といった具体的な応用事例の紹介をしている．中でも，Quantum Walk Sequencer は量子ランダムウォーク [8] を用いて音符から音符への遷移を実現するシーケンサである．この，音符から音符への遷移を量子回路によって表現するというアプローチは，メロディ生成を行う上で有用なアイデアであると考えられる．QuTune のグループでは，他にも自然言語処理の枠組みに量子コンピューティングを取り入れる量子自然言語処理（QNLP）のアプローチにより音楽生成を行うシステムについての研究 [9] なども行っている．

その他にも，Kirke によってゲート式量子コンピュータと量子アニーリングマシンを組み合わせたハイブリッドな音楽生成システム qGEN が提案されている [10]．qGEN は，ゲート式量子コンピュータによってメロディを生成する GATEMEL と量子アニーリングマシンによって与えられたメロディの伴奏を生成する qHARMONY の組み合わせによって音楽生成を行っている．

日本でも，相馬によってゲート式量子アルゴリズムに基づいて即興的に音楽生成を行う手法についての研究が行われている [11]．相馬は，連続する音符同士をエンタグルメント（量子もつれ）させることによって関連付けてメロディを生成する手法を提案している．

本章で紹介した量子コンピューティングを音楽表現に応用する試みは，どれもこの数年の間に発表されたものであり，まさに今この研究分野が動き始めようとしているところだと言える．

[*1] International Symposium on Quantum Computing and Musical Creativity, 2021 年 11 月にオンライン開催.

4. 量子コンピュータの音楽生成への応用の可能性

本章では，量子コンピュータの音楽生成への応用の可能性について議論する．これまでに記述してきたように，Quantum Music に関する試みは近年多く提案されてきている．

量子コンピュータの大きな特徴は，量子ビットの重ね合わせ状態を利用した演算を行うことができることにある．しかし，この量子ビットの重ね合わせ状態は測定によって壊れてしまい，量子ビットは古典ビットと同様の 0 か 1 のどちらかの状態に定まってしまう．量子コンピュータを用いて何かの問題を解く際には，重ね合わせ状態を利用してあらゆる状態が重ね合わされた入力を与え，所望の正解が測定される確率が高くなるように量子アルゴリズムを設計する．量子コンピュータが古典コンピュータよりも速く計算結果を得られるような問題はまだあまり多く見つかっているわけではない．現状では，因数分解や組み合わせ最適化などの特定の問題を解くためのアルゴリズムが提案されている段階であり，それらもまだ実用的な精度や規模では実現していない．

ここで，音楽生成というタスクに量子コンピュータを活用することを考える．そもそも，音楽に正解というものはなく，ただ一つの正解を求めるような処理は，少なくとも音楽の生成という文脈では必要とされない．検索などのタスクの場合には，Grover のアルゴリズムのような効率的な方法で求める音楽等を得るためのアプローチが今後提案される可能性はある．一方で，生成タスクにおいて正解を見つけるというアプローチは，いかに効率的な探索アルゴリズムが生み出されても実現できるとは考えにくい．しかし，音楽には多くの人が不正解と考えるような音の並びや組み合わせは多く存在しており，生成タスクを解く際にはそのような好ましくない結果を避けて生成できるようなアルゴリズムの実現が望ましい．これは，量子コンピュータであろうが古典コンピュータであろうが関係なく音楽生成研究において長く考えられてきたことである．

メロディを生成するタスクに量子コンピュータで取り組むことを考えたときに，量子ビットの重ね合わせ状態の表現により，存在しうるあらゆるメロディの組み合わせが同時に表現できると考えられる．存在しうるあらゆるメロディの中には，好ましくない音の組み合わせも多く存在するため，そういった組み合わせの結果が測定される確率が極端に低くなるように量子アルゴリズムを設計するというアプローチが考えられる．しかし，ここで問題となるのは，最終的に測定可能な結果はあらゆる組み合わせの中の一つとなってしまうため，せっかく量子コンピュータであらゆる組み合わせを同時に考慮していても得られるメロディは一つとなる．このことを考えると，メロディ生成というタスクにおいては，量子コンピュータを使うことによって古典コンピュータよりも優れた結果を得られるということは現時点では保証されない．

量子コンピュータと古典コンピュータはそれぞれ性質が違うため，アルゴリズムの工夫次第では量子コンピュータの導入によって効率的に結果を得られるような方法は十分に考えられる．量子コンピュータの効果的な導入の一例は，乱数を使った計算である．重ね合わせ状態にある量子ビットが 0 か 1 のどちらであるかは測定するまでは確率的にしか記述できず，その結果は正確に予測することはできない．そのため，古典コンピュータでよく用いられる疑似乱数とは違い，規則性，再現性がない真の乱数を生成することができる．既存の音楽生成アルゴリズムの多くには乱数を使用する処理が含まれており，その点で量子コンピュータを活用することは有益だと考えられる．一方で，音楽生成アルゴリズムにおいて乱数の精度はそれほど重要なものではないと考えられ，それほど重要度の高い量子コンピュータの導入事例であるとは言い難い．

本稿における初期試行として提案するメロディ生成のアルゴリズムは，量子コンピュータにおける乱数生成を用いながら，教師データに見られるような音高遷移を再現するような量子回路をデザインする手法である．これは，乱数の精度という点を除けば古典コンピュータによってより簡潔に記述可能なアルゴリズムであるが，それをあえて量子コンピュータで実装した場合の例について紹介するものである．

今後，音楽生成に量子コンピュータを応用することを考えたとき，最も考えられる活用シナリオは古典コンピュータとのハイブリッド型のアルゴリズムである．例えば，データの処理などを古典コンピュータで行い，考えられるあらゆる組み合わせの中から生成結果を絞り込んでいくような処理を量子コンピュータによって行うといったハイブリッドアルゴリズムである．現在のコンピュータがすべて量子コンピュータに置き換えられるという未来については多くの専門家も疑問に思っている．現実的には現在古典コンピュータが行う処理の一部で量子コンピュータの方がより効率的に処理できるものだけを量子コンピュータが担うという見方をされることが多い．今後しばらくの間は一部の処理を量子コンピュータが担い，その他は従来通り古典コンピュータで処理するというアプローチに関する研究が主流となっていくことが予想される．

5. 量子回路デザインのためのデータ表現

ここからは，本稿で提案する量子コンピュータによるメロディ生成の初期試行について紹介する．本稿で提案する手法は，メロディ生成のための量子回路を構成する量子ゲートを遺伝的アルゴリズムによってデザインする手法である．

表 1 音名の二進数表現

音名	二進数
C	000
D	001
E	010
F	011
G	100
A	101
B	110
R	111

量子アルゴリズムについて考える上で，どのように量子回路をデザインすれば狙い通りの入出力関係が得られるかを理解することは難しく，量子回路が複雑になればなるほどその困難さは増す．例えば，C コードの伴奏が入力として与えられた場合のメロディ生成について考えるときに，C コードの構成音であるド，ミ，ソの音が同確率で出力されるような単純な量子回路を設計することはそれほど難しくない．一方で，より複雑な条件を考えて微妙な偏りを含むような確率分布で音を出力させるような場合には量子回路の設計難易度は高くなる．

このような量子回路デザインの困難さを踏まえ，本稿では特定の音の入出力に関する条件を実現する量子回路を自動でデザインする手法について検討する．具体的には，様々な量子ゲートの組み合わせに関して入出力の試行を繰り返し，所望の出力の分布が得られるような量子回路の構成を学習するアプローチを取る．本稿の初期試行としては，量子回路を構成するゲートの組み合わせを遺伝的アルゴリズムで学習する手法を提案する．

量子回路でメロディを扱うために，データ表現の方法を決める必要がある．また，様々な量子ゲートの組み合わせによって量子回路を構成する上で，入出力の関係が所望の出力音の分布になるような量子回路を実現するために，量子ゲートそのものもデータとして数値表現する．

5.1 データの表現

本初期試行では，メロディを構成する最低限の要素のみに絞って，なるべく少ない要素について考慮した状態でメロディ生成をするような問題設定とする．メロディをなるべく単純に扱うため，音符のデータ表現は音名のみを 3 ビットの二進数で表現する形で行う．具体的には，000 を C とし，C（000）から B（110）までの 7 種類の音符と休符の R（111）を含めた計 8 種類の音名のみを扱う．単純な問題設定とするために，音価については考慮せず，全ての音符の長さが四分音符の長さであるものとする．二進数の表現と音名との対応関係を**表 1** に示す．

3 ビットの二進数の各桁をそれぞれ一つの量子ビットに対応させるものとし，3 量子ビットの量子回路によって入力の音に対する後続音を生成するような問題設定とする．

表 2 量子ゲートの数値表現

ゲートの種類	数値表現
H ゲート（レジスタ 0）	0
H ゲート（レジスタ 1）	1
H ゲート（レジスタ 2）	2
X 軸周りの回転ゲート：$\pi/4$（レジスタ 0）	3
X 軸周りの回転ゲート：$\pi/4$（レジスタ 1）	4
X 軸周りの回転ゲート：$\pi/4$（レジスタ 2）	5
CX ゲート（レジスタ 0 → 1）	6
CX ゲート（レジスタ 0 → 2）	7
CX ゲート（レジスタ 1 → 2）	8
CX ゲート（レジスタ 1 → 0）	9
CX ゲート（レジスタ 2 → 0）	10（A）
CX ゲート（レジスタ 2 → 1）	11（B）
CCX ゲート（レジスタ 0,1 → 2）	12（C）
CCX ゲート（レジスタ 0,2 → 1）	13（D）
CCX ゲート（レジスタ 2,1 → 0）	14（E）
ゲート無し	15（F）

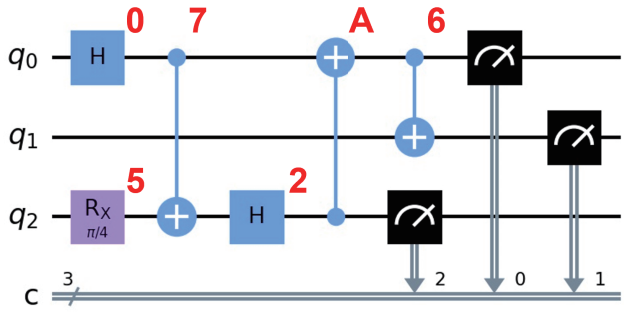

図 1 「F0572AF6」で表現される量子回路の例

5.2 量子ゲートの数値表現

量子回路を構成する量子ゲートには様々な種類がある．本稿では，なるべく単純な問題設定でメロディ生成を実現するために，基本的なゲートを 4 種類選び，その配置パターン（どの量子ビットに対して作用させるか）を含めた 16 通りのゲート選択の方法を数値表現する．本稿で扱う量子回路は，16 通りの配置で選択されたゲートを 8 個組み合わせることで構成する．16 通りの中にはゲートを配置しない状態も含めており，0 個から 8 個のゲートを組み合わせた量子回路を表現可能としている．16 通りの量子回路の配置方法を 16 進数の数値で表現するものとし，その数値表現は**表 2** に準じるものとする．

表 2 の量子ゲートの数値表現を用いると，8 桁の 16 進数の数値で量子回路を表すことができる．例えば，「F0572AF6」の回路図は**図 1** のようになる．表 2 の対応表によると，F は 15 のゲート無しに対応するため，この場合，6 個のゲートからなる量子回路となる．0 はレジスタ 0（図 1 の q_0 にあたるレジスタ）に対する H ゲート（アダマールゲート）を表し，これにより量子ビットは重ね合わせ状態となる．5 はレジスタ 2 に対する X 軸周りの回転ゲートで，その回転角度は $\pi/4$ としている．7，A，6 はすべて CX ゲート（制

表 3 図 1 の量子回路に，|000⟩（C）を入力した際の測定結果（200 ショット分）

出力値	測定回数
000（C）	48
001（D）	0
010（E）	0
011（F）	48
100（G）	43
101（A）	0
110（B）	0
111（R）	61

御 NOT ゲート）で，これによりレジスタ間の相互作用を実現するようなゲートとなる．

量子ゲートの種類はここに挙げたもの以外にも数多く存在しており．他のゲートに対しても番号を割り当てさえすれば同じ枠組みでさらに複雑な量子回路も表現可能である．しかし，本稿ではあくまでも初期試行としてなるべく単純に問題に取り組むことを重視しているため，ここに挙げた基本的なゲートのみを使用することとする．また，量子の重ね合わせ状態を利用しない場合，0 と 1 のみを用いた古典コンピュータによる演算と同じ結果が得られ，量子回路を使用する意味がなくなってしまう．そこで，レジスタ 0 から 2 の入力には，すべて H ゲートを 1 度作用させた状態にし，そこに 8 桁の 16 進数の数値で表されるその他のゲートを追加していくことにする．

5.3 量子回路による後続音の生成

以上の音名に関するデータ表現と量子ゲートの数値表現を組み合わせることで，入力された音に対して後続音を生成するという問題を量子回路へのデータの入出力で表現する．メロディ生成の手順としては，まず任意の第一音を決めて量子回路に入力し，最終的な量子ビットの状態を測定することで後続音を生成する．さらに，後続音を再度入力データとして，その音名に対応する量子回路に入力し，さらなる後続音を生成するという手順を繰り返すことによってメロディを生成する．

図 1 の量子回路に入力音 C に対応する量子ビット，|000⟩ を入力したときに得られる測定結果（200 ショット分）を **表 3** に示す．ここで，実際に後続音を生成する際にはすべての入力にまず H ゲートが 1 度作用された状態で処理が行われることに注意されたい．表 3 の結果は H ゲートを作用させず，図 1 の量子回路に，|000⟩（C にあたるデータ）をそのまま入力した場合の結果である．

量子回路は，入力音の種類毎に異なるものを用意する．例えば，入力音が C の場合には C の量子回路を使って後続音を生成し，入力音が D の場合には D の量子回路を使って後続音を生成する．それぞれの入力音に対応する量子回路のデザイン方法については次節で紹介する．

6. 遺伝的アルゴリズムを用いた量子回路のデザイン

量子回路をデザインすることは量子アルゴリズムを決定することに相当し，いかに設計するかが生成結果を左右する．ここで，量子回路をデザインする方針としては様々なアプローチが考えられる．例えば，すべてのレジスタに H ゲートを一つずつ配置するような量子回路を作成すれば，すべての出力の組み合わせが重ね合わされた状態を表現でき，完全にランダムな出力が得られるような回路ができあがる．

回路を構成するゲートを手動で決めることも考えられるが，その場合，一度の量子回路のデザインにより生成結果の大まかな方向性が固定されてしまうことになり柔軟性に欠ける．そこで，本稿では，量子回路のデザインを教師データに基づいて柔軟に変更できるような手法を検討する．具体的には，所望の出力音の分布を表す教師データを用意し，その分布を再現できるような量子ゲートの組み合わせを探索する．これにより，実現したい入出力関係が明確な状態で，それを再現するような量子回路のデザインを行うという明確な基準を設けることができる．量子ゲートの組み合わせを探索する手法としては，遺伝的アルゴリズムを採用する．遺伝的アルゴリズムを用いて，量子回路への入力に対する出力と，教師データの出力音分布との差が小さくなるような学習を行う．ここで，遺伝的アルゴリズムの処理自体は古典コンピュータによって処理を行い，量子コンピュータによる処理は入力音に対する後続音の生成の部分のみであることに注意されたい．また，実際にはこれらの学習の処理をすべて古典コンピュータによる量子回路シミュレータを用いて実行する．最終的な量子回路を用いてメロディを生成する手順のみで実際の量子コンピュータを利用することを想定している．量子回路のデザインのプロセスについても実際の量子コンピュータを使うことが考えられるが，試行回数が多いため，現状の量子コンピュータの利用形態では現実的な処理時間で学習ができない．

6.1 教師データの準備

教師データは既存の楽曲の音名列を基に作成する．提案アルゴリズムで用いる音の種類は 8 種類で，すべて四分音符の長さとするため，学習する楽曲もハ長調の四分音符のみで構成される単純なメロディの楽曲とする．具体的に，学習する楽曲として選んだものは，「きらきら星」，「チューリップ」，「カエルのうた」の 3 曲分のメロディである．

例えば，「きらきら星」の冒頭部のメロディから教師データを作成する場合，メロディは「C → C → G → G → A → A → G → R」であり，C の音の後続音は C か G であり，その他の音への遷移はないものとする．これを再現する理想的な量子回路は，C（000）の入力に対して C（000）と

表 4 教師データ（音名の遷移確率）

	後続音							
	C	D	E	F	G	A	B	R
C	0.10	0.38	0	0	0.10	0	0	0.43
D	0.33	0.14	0.38	0	0	0	0	0.14
入 E	0.04	0.40	0.20	0.12	0.04	0	0	0.20
力 F	0	0	0.58	0.33	0.08	0	0	0
音 G	0	0	0.17	0.17	0.28	0.22	0	0.17
A	0	0	0	0	0.57	0.73	0	0
B	0	0	0	0	0	0	0	0
R	0.50	0	0.11	0.11	0.28	0	0	0

G（100）がそれぞれ 50%の確率で測定されるような量子回路となる．このような入出力の関係は，例えば第 2 レジスタに H ゲートが一つ挿入された量子回路によって実現可能である．実際に学習する教師データはもう少し複雑な出力の分布となるため，手動での量子回路のデザインではなく，遺伝的アルゴリズムによる自動でのデザインを行う．

教師データとなるのは音名の遷移確率であり，デザインする量子回路はバイグラムによる後続音の生成モデルに相当することとなる．実際の教師データとしては，3 曲分の音名遷移を一つの行列で表現したものとする．教師データとなる音名の遷移確率を**表 4** に示す．この音名の遷移確率によると，A の音が入力された場合，後続音としては再度 A が生成される確率が 0.73 と最も高く，次に G が生成される確率が 0.57 で，それ以外の音の確率は 0 となる．

この教師データは，扱う曲の数を増やすことですぐに拡張できるものであるが，本稿におけるメロディ生成は 8 種類の音のみを扱い，すべての音を四分音符で鳴らすという単純な問題設定であるため，扱うことができる楽曲の種類は多くない．音価に関する制約を取り除き，扱うことができる音名の種類を増やすことで，同じ枠組みのままでより複雑なメロディの分布も学習可能ではあるが，本稿では検討しない．

6.2 遺伝的アルゴリズムの詳細

遺伝的アルゴリズムによって学習するのは，5.2 節で紹介した量子ゲートの組み合わせを表現した 8 桁の 16 進数の数値列である．ランダムに初期化した 8 桁の 16 進数の数値列を個体とみなし，各桁が遺伝子に相当するものとする．各世代 1000 個体をトーナメント選択，二点交叉，突然変異のステップにより適応度の高い個体が優先的に選択されるように 100 世代分処理を繰り返す．適応度は，表 3 に示したような 200 ショット分の出力結果をショット数で割った出力分布と，学習したい所望の出力音分布（表 4）との間の平均二乗誤差によって算出する．

量子回路を構成するゲートがない場合の遺伝子配列は「FFFFFFFF」となり，この状態だと入力の量子ビットにはそれぞれ H ゲートが一度ずつ作用する状態となり，すべての状態の重ね合わせ状態となる．この状態を基に，遺伝子配列が変わって量子ゲートが挿入されていくことで，教師データに近づくように偏った出力が得られる量子回路のゲート構成を学習する．出力の分布が教師データに近づくほど適応度が高いものとすることで，所望の出力分布が得られるような量子回路のデザインを自動で行うことができる．

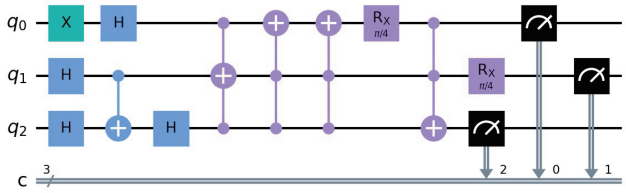

図 2 遺伝的アルゴリズムによってデザインされた入力音が D の場合の量子回路

表 5 学習した量子回路による後続音の生成結果（100 ショット）

		生成された後続音（測定回数）							
		C	D	E	F	G	A	B	R
	C	19	20	0	0	27	0	0	34
	D	31	22	14	4	0	4	0	25
量	E	8	38	13	18	0	0	11	12
子	F	0	0	56	44	0	0	0	0
回	G	4	1	24	30	12	19	5	5
路	A	0	0	0	0	60	40	0	0
	B	12	12	10	14	10	14	10	18
	R	43	0	7	6	35	0	5	4

表 4 の教師データを用いて遺伝的アルゴリズムによってデザインした量子回路の一つを**図 2** に示す．図 2 の量子回路は入力音が D の場合の量子回路であるため，レジスタ 0 に最初に X ゲートが配置されている．これにより，入力 $|000\rangle$ が $|001\rangle$ に変化した状態から始まって，すべての量子ビットに H ゲートが作用してその後の処理が行われるようになっている．図 2 の量子回路では，同じ CCX ゲートが二連続で並んでいる箇所などの意味のないゲート配置も見られるが，この回路から得られる出力は，教師データ $[0.33, 0.14, 0.38, 0, 0, 0, 0, 0.14]$ に近い分布となる．

量子回路は入力音の数だけ用意しており，遺伝的アルゴリズムによって全部で 8 種類の回路がデザインされることとなる．入力音が D の場合以外の量子回路のデザイン結果の一覧については巻末に付録としてまとめて示す．

6.3 学習した量子回路による後続音の生成

遺伝的アルゴリズムにより教師データを学習した量子回路による後続音の生成を行った．各入力音に対応する 8 種類の回路をそれぞれ 100 ショットずつ実行して得られた出力を**表 5** に示す．後続音を生成する際には 1 ショットのみ実行し，測定された出力が後続音となる．実行するたびに結果は異なるが，概ね表 5 に近い分布で後続音が得られる．

表 6 生成されたメロディ（音名列）の例

第一音	生成されたメロディ（音名列）
C	C G F F F E B G A G D C C D E D
C	C D D D R E D R C C G F E B D D
G	G D D C G A G E F E B C R E R C
G	G F F F E R E D D E R G G E R C

表 4 と表 5 を比較することで，学習した量子回路による後続音の出力の分布がどの程度教師データに近づいているかを確認することができる．教師データでは，B への遷移確率はどの音の場合も 0 であったが，得られた量子回路では B への遷移が起こることがあった．また，入力音 B の量子回路は学習すべき出力の分布がすべて 0 となっているため，すべての出力が測定されるような量子回路となった．

量子コンピュータの場合，実行するごとに結果が変わってしまうため，厳密に分布を再現することは難しく，学習時の試行では教師データと近い分布を得られたとしても，再度試行した際にはその時の出力分布が再現されるわけではない．

7. メロディの生成結果

これまでに記述した手法でデザインした量子回路で，実際にメロディを生成した．メロディの生成は，まず最初の一音を選び，その音名に対応する量子回路に入力した際に得られる出力を後続音とし，さらにその後続音に対応する量子回路にその音を入力した際に得られる出力を第三番目の音とする，といった手順を繰り返すことで行う．第一音目の音名と何音分のメロディを生成するかに対応する繰り返し回数は任意に決めることができる．**表 6** に，最初の一音を C または G として計 16 音 4 小節分のメロディを生成した結果の例（4 試行分）を示す．後続音は確率的に測定されるため実行する度に異なる結果が得られる．この結果は Qiskit の量子回路シミュレータである Qasm Simulator によって得られた結果である．

生成されたメロディには，教師データには含まれない B の音への遷移が含まれるケースがあった．一方で，教師データに含まれる音の遷移が再現されている箇所も見受けられ，学習対象となったメロディの雰囲気が感じられるケースがあった．

次に，実際の量子コンピュータを用いてメロディの生成を行った．実際の量子コンピュータの場合，エラーが起こることがあるため，エラーによって表 5 では 0 となっているような音の遷移（F → D や C → B 等）も起こることがある．使用した量子コンピュータは，IBM 社がクラウド上で提供している IBM Q の中の 5 量子ビットまでの演算が可能な ibm_bogota である．実際の量子コンピュータで生成したメロディを**図 3** に示す．第一音は C 及び G として 4 小節のメロディを 2 つ生成した．

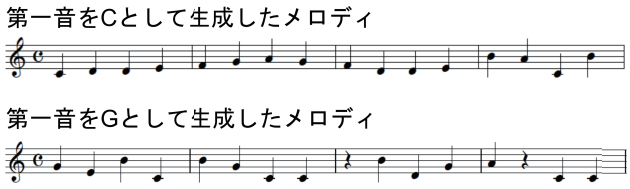

図 3 実際の量子コンピュータによるメロディの生成例

生成のための実行時間は，その時点での量子コンピュータのタスク実行までの待ち時間に依存するが，上記の 16 音のメロディ生成に要した時間は 20〜30 分程度であった．

8. まとめ

本稿では，量子コンピュータによるメロディ生成の試みとして，量子コンピュータの音楽生成への応用の可能性について議論するとともに，その初期試行として，遺伝的アルゴリズムを用いた量子回路デザインの手法を提案した．特定の楽曲で学習した量子回路を用いて，新たなメロディを生成した結果を示した．同様のメロディ生成は，結果のランダム性の違いを除けば，古典コンピュータのみでも実現可能なものとなっており，量子コンピュータの特性を活かしたような処理とはなっていない．この点については今後さらなる可能性を探求していきたいと考えている．

また，今回のメロディ生成では，問題設定をなるべく単純にするために，使用可能な音の種類や音の長さ，量子ゲートの種類や数などを絞って処理を行った．この制約については容易に拡張可能であるが，それは複雑な表現を実現するための拡張となり，量子コンピュータの可能性についての議論を目的とした本稿ではあまり重視しなかった．

量子コンピューティングの技術そのものが発展途上にあり，今後，量子回路を用いない量子プログラミングの方式も提案されていくことが予想される．量子コンピュータの扱い方の変化に応じてアプリケーションのあるべき姿も大きく変わっていくと考えられる．今後，引き続き量子コンピュータの音楽生成への応用の可能性について探求を続けていきたい．それにより，これまでの古典コンピュータでは実現できなかったような音楽生成における表現の拡張等を目指していきたい．

謝辞 本研究の一部は JSPS 科研費 JP19K20301 の助成を受けたものである．

参考文献

[1] Gambetta, J.: Expanding the IBM Quantum roadmap to anticipate the future of quantum-centric supercomputing. https://research.ibm.com/blog/ibm-quantum-roadmap-2025.

[2] Kirke, A., Shadbolt, P., Neville, A., Antoine, A. and Miranda, E.: Q-Muse: A quantum computer music system designed for a performance for orchestra, electronics and live internet-connected photonic quantum computer (2014).

[3] Clemente, G., Crippa, A., Jansen, K. and Tüysüz, C.: New Directions in Quantum Music: concepts for a quantum keyboard and the sound of the Ising model, *arXiv preprint arXiv:2204.00399* (2022).
[4] Weimer, H.: Listen to Quantum Computer Music. http://www.quantenblog.net/physics/quantum-computer-music.
[5] Miranda, E. R., Coecke, B. et al.: QuTune Project. https://iccmr-quantum.github.io/.
[6] Miranda, E. R.: Quantum Computer: Hello, Music!, *Handbook of Artificial Intelligence for Music*, Springer, pp. 963–994 (2021).
[7] Miranda, E. R. and Bask, S. T.: Quantum Computer Music: Foundations and Initial Experiments, *arXiv preprint arXiv:2110.12408* (2021).
[8] Aharonov, Y., Davidovich, L. and Zagury, N.: Quantum random walks, *Physical Review A*, Vol. 48, No. 2, p. 1687 (1993).
[9] Miranda, E. R., Yeung, R., Pearson, A., Meichanetzidis, K. and Coecke, B.: A Quantum Natural Language Processing Approach to Musical Intelligence, *arXiv preprint arXiv:2111.06741* (2021).
[10] Kirke, A.: Programming gate-based hardware quantum computers for music, *Musicology*, No. 24, pp. 21–37 (2018).
[11] 相馬聡文: 量子的アルゴリズムに基づく即興的音楽生成の可能性に関する一考察, 日本音楽即興学会 第 13 回大会 (2022).

付　　録

提案手法でデザインされた量子回路の例

本稿で述べた手法によって実際にデザインした量子回路を以下に示す（入力音 D の場合については図 2 を参照のこと）.

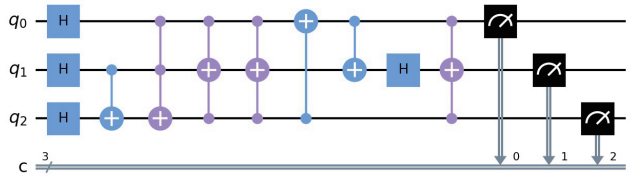

図 A·1　提案手法でデザインされた量子回路（入力音 C の場合）

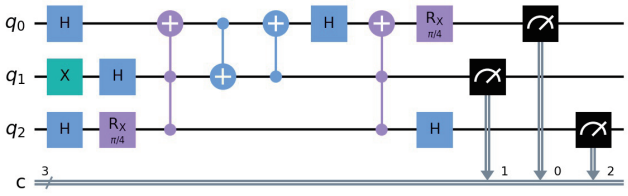

図 A·2　提案手法でデザインされた量子回路（入力音 E の場合）

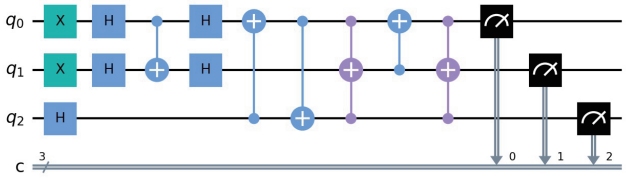

図 A·3　提案手法でデザインされた量子回路（入力音 F の場合）

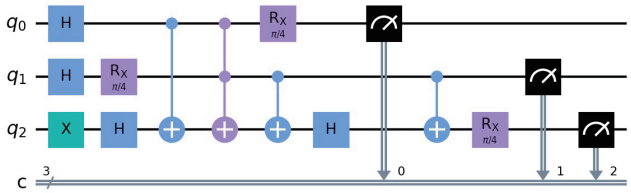

図 A·4　提案手法でデザインされた量子回路（入力音 G の場合）

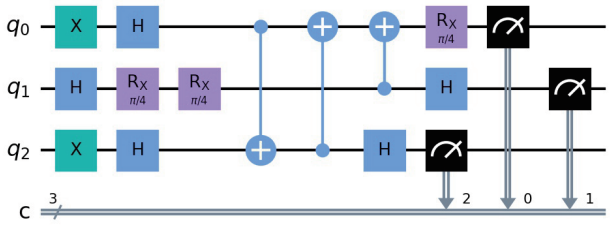

図 A·5　提案手法でデザインされた量子回路（入力音 A の場合）

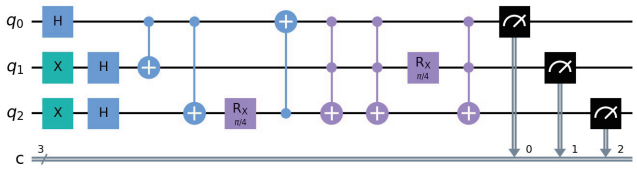

図 A·6　提案手法でデザインされた量子回路（入力音 B の場合）

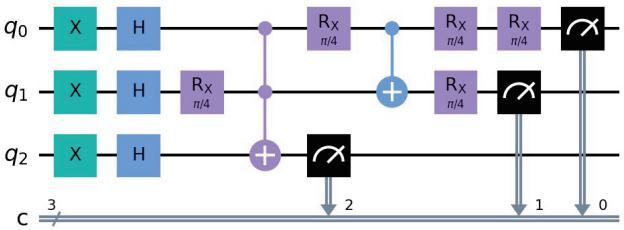

図 A·7　提案手法でデザインされた量子回路（入力音 R の場合）